JOURNAL OF COMPUTATIONAL PHYSICS 63, 67–84 (1986)

CONKUB: A Conversational Path-Follower for Systems of Nonlinear Equations

R. MEJIA

Intramural Research, National Heart, Lung, and Blood Institute and Mathematical Research Branch, National Institute of Arthritis, Diabetes, and Digestive and Kidney Diseases, National Institutes of Health, Bethesda, Maryland 20892

Received November 6, 1984; revised February 27, 1985

We describe an interactive computer program to trace solutions of systems of nonlinear algebraic equations and illustrate its application to solve several difficult problems. Turning points and bifurcations are located and solution branches are identified and traced interactively. Of special interest is its application to study solutions of large, sparse systems of nonlinear equations that result from the discretization of boundary value problems. Such systems arise in the description of physical, biological, and chemical phenomena. As an example, we show a model of urine formation in the mammalian kidney [13], where path-following in a subspace makes tracing the solution surface possible.

1. INTRODUCTION

CONKUB permits the study of relatively large systems of nonlinear algebraic equations, $F(x, \alpha) = 0$, with vector of functions F, unknowns x and parameters α. Such systems often arise from the discretization of nonlinear differential equations that describe physical, biological, and chemical phenomena. An example is the multipoint boundary value problem described by Mejia and Stephenson [13]. We wish to study such a system of convection–diffusion equations as a function of individual membrane parameters because, in general, there exist multiple solutions to these equations, and their number and time stability changes with variations in the parameters [14].

CONKUB consists of

(1) a driver that controls program flow, allowing (and prompting) the user to set data, parameters and program function interactively;

(2) a path-following procedure to compute points along a solution curve of an underdetermined system of equations;

(3) a procedure to identify and record turning and branch points.

The path-following procedure is based on methodology due to Keller [10] and Kubiček [11]. Turning points and simple bifurcations are treated as by Crandall and Rabinowitz [5], and Bunow and Kernevez [2]; a scheme for the treatment of

0021-9991/86 $3.00

multiple branching is also given. Methodology for following solutions to nonlinear systems has also been described, for example, by Chow, Mallet-Paret, and Yorke [3], Peitgen and Prüfer [20], Watson [28], Allgower and Georg [1], Moré and Cosnard [17], Rheinboldt [22], Georg [7], Peitgen and Schmitt [21], Rheinboldt and Burkhardt [23], Morgan [19], and Kearfott [9]. Algorithms for continuation of solutions include those by Jürgens, Peitgen, and Saupe [8], Moré and Cosnard [18], Watson and Fenner [29], and Rheinboldt and Burkhardt [24]. A computer program for the bifurcation analysis of autonomous systems of ordinary differential equations has been developed by Doedel [6], and a recent version permits a limited bifurcation analysis of algebraic systems.

To use CONKUB the user provides a subroutine to evaluate the functions F and approximations to the Jacobian and Hessian matrix when needed. CONKUB then facilitates solution in several ways:

(1) Calculation is interactive, so that the result of a command may be considered before the next command is issued.

(2) The choice of the parameter (or unknown) to be varied at any step is usually made by CONKUB, but may be specified by the user.

(3) Solution may proceed in either a positive or negative direction in the variable being followed. Hence at any step one may proceed in the current direction, turn, or target to a solution with a given value of this variable. A list of commands and their description is given in Appendix A.

(4) Control parameters and bounds set by the user may be viewed and modified during a calculation.

The computer program, which consists of approximately 2000 FORTRAN statements, will appear elsewhere and is available from the author upon request (preferably via computer mail to Mejia @ MIT-MULTICS.ARPA).

In Section 2 we describe the path-following algorithm used and the treatment of turning points and bifurcations. In Section 3 we describe a method for path-following in a subspace that we use to treat large, sparse problems. In Section 4 we give three examples that illustrate the method. The first example illustrates the use of CONKUB to follow a curve loosely sometimes (to reduce computation) and very closely other times (to obtain a solution accurately). Appendix C shows how this is done. The second example illustrates the ability to identify and trace multiple tangential arcs. The third example shows the ability to trace solutions for a relatively large problem that has been partitioned as shown Section 3. Here the method is used to obtain solutions for the kidney model described in Appendix B.

2. Continuation Method

Given a system of nonlinear algebraic equations

$$F(x, \alpha) = 0, \qquad F: \quad R^m \times R^q \to R^m \tag{2.1}$$

with unknowns x, parameters α and F sufficiently smooth, we use the assumed continuity and differentiability of F in x and α to derive the differential equations

$$\frac{dx}{d\alpha_l} + (F_x)^{-1}\frac{\partial F}{\partial \alpha_l} = 0, \qquad x(\alpha^0) = x^0, \tag{2.2}$$

with $F_x \equiv \partial F_i/\partial x_j$, $\partial F/\partial \alpha_l \equiv (\partial F_1/\partial \alpha_l, \ldots, \partial F_m/\partial \alpha_l)^T$, $1 \leqslant i, j \leqslant m$,

$$F(x^0, \alpha^0) = 0, \qquad 1 \leqslant l \leqslant q, \qquad \text{and} \qquad \alpha_k \text{ fixed for } k \neq l.$$

If F_x is nonsingular in a neighborhood of $(x(\alpha^*), \alpha^*)$, then the vector $x(\alpha^*)$ obtained by integrating Eqs. (2.2) is a solution of Eqs. (2.1). However, when F_x is singular, which occurs at a turning or branching point, or when it is nearly singular, this scheme fails. To avoid this problem and to maximize numerical stability, we use an algorithm due to Kubiček [11] to exchange the role of α_l and a dependent variable in order to solve Eqs. (2.1) for (x, α_l). The algorithm is as follows: Parametrize with respect to the arc length s of the solution locus and differentiate F with respect to s to obtain the system of equations

$$\frac{dF_i}{ds} = \sum_{j=1}^{m} \frac{\partial F_i}{\partial x_j}\frac{dx_j}{ds} + \frac{\partial F_i}{\partial \alpha_l}\frac{d\alpha_l}{ds} = 0, \qquad i = 1, 2, \ldots, m, \quad 1 \leqslant l \leqslant q. \tag{2.3}$$

The parameter s is determined as arc length along the solution curve in R^{m+1} by the equation

$$\sum_{j=1}^{m} \left(\frac{dx_j}{ds}\right)^2 + \left(\frac{d\alpha_l}{ds}\right)^2 = 1. \tag{2.4}$$

Equation (2.3) may be considered as a system of m equations in the $m+1$ unknowns $x_1, x_2, \ldots, x_m, \alpha_l$, and the system may be solved with respect to any one of the variables. Let x_k be the independent variable, and let the matrix

$$\Gamma_k = \begin{bmatrix} \dfrac{\partial F_1}{\partial x_1} & \cdots & \dfrac{\partial F_1}{\partial x_{k-1}} & \dfrac{\partial F_1}{\partial x_{k+1}} & \cdots & \dfrac{\partial F_1}{\partial x_{m+1}} \\ \vdots & & & & & \vdots \\ \dfrac{\partial F_m}{\partial x_1} & \cdots & \dfrac{\partial F_m}{\partial x_{k-1}} & \dfrac{\partial F_m}{\partial x_{k+1}} & \cdots & \dfrac{\partial F_m}{\partial x_{m+1}} \end{bmatrix} \tag{2.5}$$

be nonsingular, where we have written $x_{m+1} = \alpha_l$ for consistency. We can then solve the system (2.3) to obtain the equations

$$\frac{dx_i}{ds} = \beta_i \frac{dx_k}{ds}, \qquad i = 1, 2, \ldots, k-1, k+1, \ldots, m+1. \tag{2.6}$$

Substitution of (2.6) into (2.4) yields the result

$$\frac{dx_k}{ds} = \pm\left(1 + \sum_{\substack{l=1 \\ l\neq k}}^{m+1} \beta_l^2\right)^{-1/2}, \tag{2.7}$$

with the sign of (2.7) chosen to preserve the orientation along the curve. Equations (2.6) and (2.7) are integrated explicitly using a variable order Adams–Bashforth multistep method, and the truncation error incurred is corrected by applying Newton's method to remain within a specified distance of the solution curve.

At each value calculated along the solution curve a change of sign (or a change of direction near zero) in the determinant of F_x indicates the possible existence of a singularity. We use bisection to obtain (x^*, α^*) such that $\det F_x(x^*, \alpha^*) \simeq 0$. Other criteria are given by Allgower and Georg [1], Jürgens *et al.* [8] and Moore and Spence [16].

Turning points are thus readily identified. Our technique for identifying and tracing arcs at branch points is based on that of Crandall and Rabinowitz [4, 5], Keller [10] and the implementation of Bunow and Kernevez [2]. A treatment of multiple bifurcations is given by Kearfott [9]. Our approach has been to treat all bifurcation points as if simple.

Let the dimension of the null space of F_x at (x^*, α^*), $\dim N(F_x^*) = 1$, then $N((F^*)^T) = \text{span}\,\psi_1$ and $|\psi_i^T F_{\alpha_i}| \leqslant \delta$ with δ small indicates (x^*, α^*) is a bifurcation point. Since we know the tangent of the original branch at a simple bifurcation, we use the bifurcation condition, $F_{\alpha_i}^* \in R(F_x^*)$, to obtain the tangent of the bifurcating branch. A small step in each direction yields points that are corrected onto the bifurcating branch [2] using Newton's method. At multiple branch points we do an interactive search.

3. Path-Following in a Subspace

It has previously been shown [12, 13] that certain multipoint boundary value problems can be discretized and partitioned for iterative solution (see Appendix B for an example). Consider such a system of differential equations which has been discretized using finite differences to form a set of nonlinear algebraic equations

$$F(\gamma; \alpha) = 0, \qquad F: \quad R^{n+m} \times R^q \to R^{n+m}, \tag{3.1}$$

with unknowns γ and parameters α. Equations (3.1) may be partitioned and written as

$$F_l(\gamma_l, \gamma_M; \alpha) = 0, \qquad F_l: \quad R^{n_l+m} \times R^q \to R^{n_l},$$

$$l = 1, 2, \ldots, L, \qquad n_l \geqslant 3,$$

$$F_M(\gamma_1, \gamma_2, \ldots, \gamma_L, \gamma_M; \alpha) = 0, \qquad F_M: \quad R^{n+m} \times R^q \to R^m,$$

$$\sum_{l=1}^{L} n_l = n \gg m, \tag{3.2}$$

with vectors $\gamma_l \in R^{n_l}$, $\gamma_M \in R^m$ and $\alpha = (\alpha^1, \alpha^2, \ldots, \alpha^q) \in R^q$.

To obtain solutions of Eqs. (3.1) as a function of a model parameter α^h, we solve the analog of Eqs. (2.3) and (2.4); namely,

$$\begin{aligned} \frac{\partial F_M}{\partial \alpha^h}\dot{\alpha}^h + \frac{\partial F_M}{\partial \gamma_M}\dot{\gamma}_M &= 0, \\ \|\dot{\gamma}_M\|^2 + (\dot{\alpha}^h)^2 &= 1, \\ F_l(\gamma_l, \gamma_M; \alpha) &= 0, \qquad l = 1, 2, \ldots, L, \end{aligned} \tag{3.3}$$

with initial conditions $\gamma(0) = \gamma^0$, $\alpha(0) = \alpha^0$ and for $F(\gamma^0; \alpha^0) = 0$. Recall that s has been defined as arc length so that $\gamma_M = \gamma_M(s)$, $\alpha^h = \alpha^h(s)$ and $\dot{\alpha}^h = d\alpha^h/ds$.

Now if $(\gamma(s), \alpha(s))$ is a solution of (3.3), then for each s $(\gamma(s), \alpha(s))$ solves (3.1). Conversely, if $(\gamma(s), \alpha(s))$: $s_a < s < s_b$ is a branch of solutions of (3.1) (i.e., $(\gamma(s), \alpha(s))$ is not an isolated solution), then $(\gamma(s), \alpha(s))$ solves (3.3) with suitable initial conditions.

Suppose that $(\hat{\gamma}; \hat{\alpha})$ is a turning or branching point of (3.3), so that $\det\{(\partial F_M/\partial \gamma_M)(\hat{\gamma}; \hat{\alpha})\} = 0$. We exchange the role of a dependent variable γ_M^j with the parameter α^h, so that for some j, $1 \leqslant j \leqslant m$,

$$\frac{\partial F_M}{\partial \gamma_M}(\hat{\gamma}_M^1, \hat{\gamma}_M^2, \ldots, \hat{\gamma}_M^{j-1}, \hat{\alpha}^h, \hat{\gamma}_M^{j+1}, \ldots, \hat{\gamma}_M^m, \hat{\alpha}^1, \hat{\alpha}^2, \ldots, \hat{\alpha}^{h-1}, \hat{\gamma}_M^j, \hat{\alpha}^{h+1}, \ldots, \hat{\alpha}^q)$$

is invertible. In this manner we proceed along a path on the solution surface of (3.3) and hence of (3.1). Variation of h, $1 \leqslant h \leqslant q$, permits study of the solutions as a function of each of the parameters α.

4. Examples

The first example is due to Watson [28]. Consider the homotopy map

$$r(v, \alpha) = \alpha f + (1 - \alpha)(v - v^0) \tag{4.1}$$

starting at $(v^0, 0)$ and follow the zero curve until $\alpha = 1$. Let f: $R^3 \to R^3$ be defined by

$$f_k = v_k - \exp\left[\cos\left(k \sum_{i=1}^{3} v_i\right)\right], \qquad k = 1, 2, 3 \tag{4.2}$$

and $v^0 = 0$.

Figures 1, 2, and 3 show each component of v plotted against the parameter. Note the four turning points for $0 \leqslant \alpha \leqslant 1$, and that large steps are taken for most of

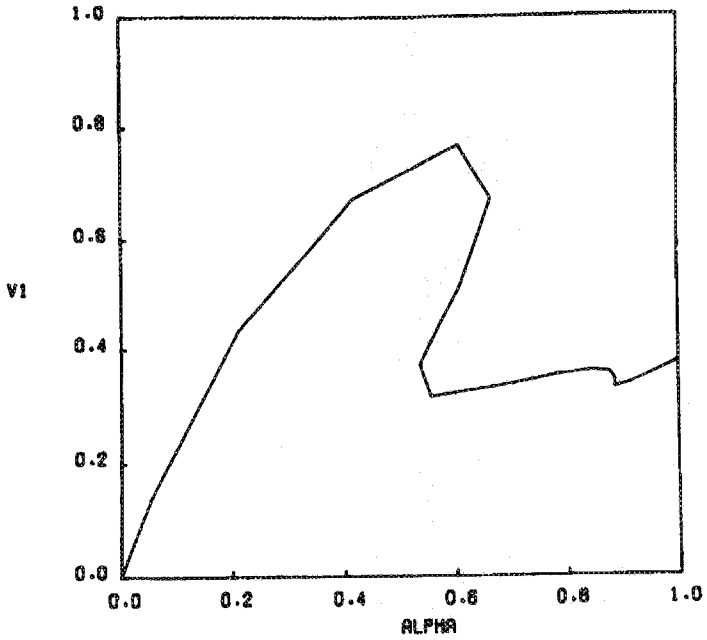

FIG. 1. The first component v_1 of the solution of Eq. (4.1) is plotted as a function of the parameter.

the trajectory. Only near $\alpha = 1$ are small steps required. A portion of the log of the session to obtain these figures is given in Appendix C. It includes the initial guess and bounds specified, the solutions calculated and interactive viewing and changes of parameters.

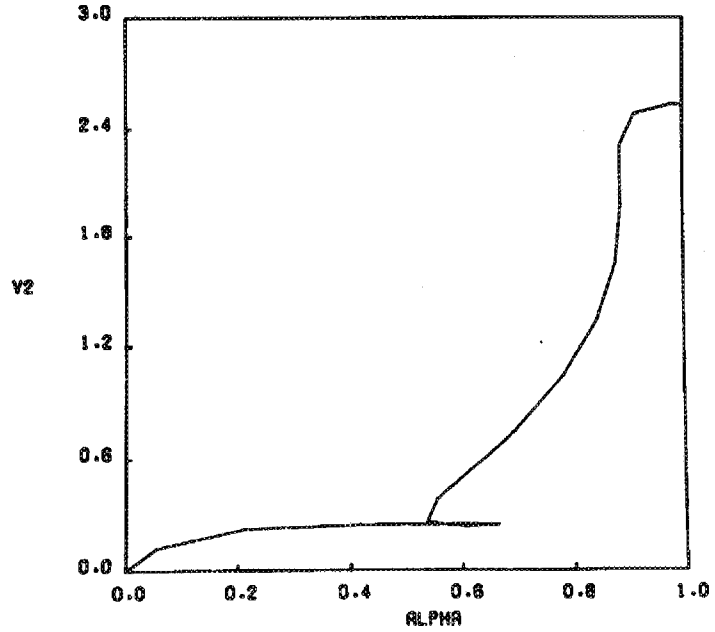

FIG. 2. The second component of v is plotted versus the parameter. Note the abrupt turn at $\alpha \doteq 0.67$.

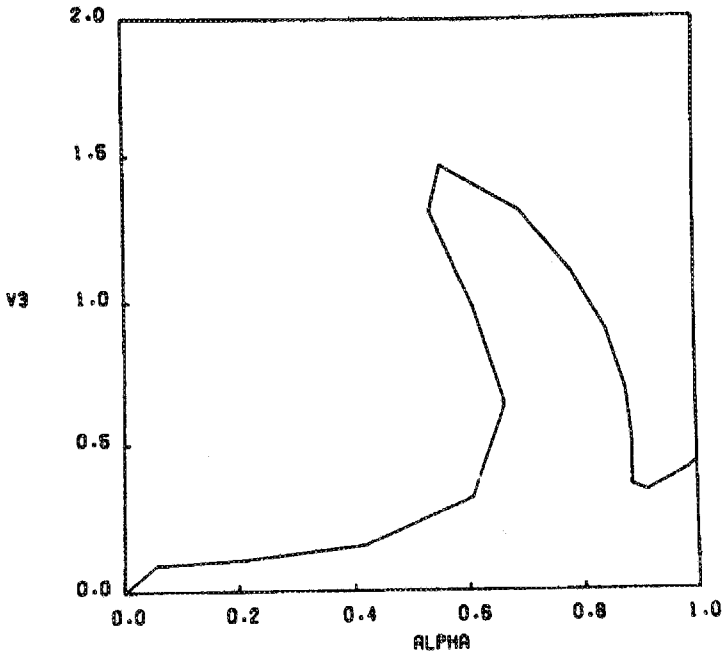

FIG. 3. The third component of the solution is plotted against the parameter.

In the second example due to Kearfott [9] we consider the map

$$r(x, \alpha) = \alpha g(x) + (1 - \alpha)\, g^0(x). \tag{4.3}$$

$g: R^3 \to R^3$ is defined by $g(x) = Ax - f(x)$, and $g^0: R^3 \to R^3$ is defined by $g^0(x) = -Ax$ for $f_i = x_i^3$, $i = 1, 2, 3$, and

$$A = 16 \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}. \tag{4.4}$$

Figures 4 and 5 show the four solution arcs that intersect at $\alpha = 0.5$. We start at solution (0, 0, 0, 0) and proceed to identify and trace the tangent solutions by tracing each branch detected. A rigorous, more time consuming procedure might be invoked instead (see for example [6, 9, or 20]).

The third example, due to Mejia and Stephenson [15], is a model of solute and water flow in the mammalian kidney that is described in Appendix B. The discretized model consists of several hundred equations whose solutions we seek as a function of membrane parameters. Continuation of the solution is carried out in a subspace with dimension $m = 53$ using the procedure described in Section 3.

Figure 6 is a schematic diagram of the model. Figure 7 shows the total urine concentration (the outflow of tube *CD* in Fig. 6) as a function of the maximum rate of salt transport out of the thick ascending limb of Henle (tube *AHL* in the outer medulla) of the long nephrons. Note that the hysterisis loop traced by the concen-

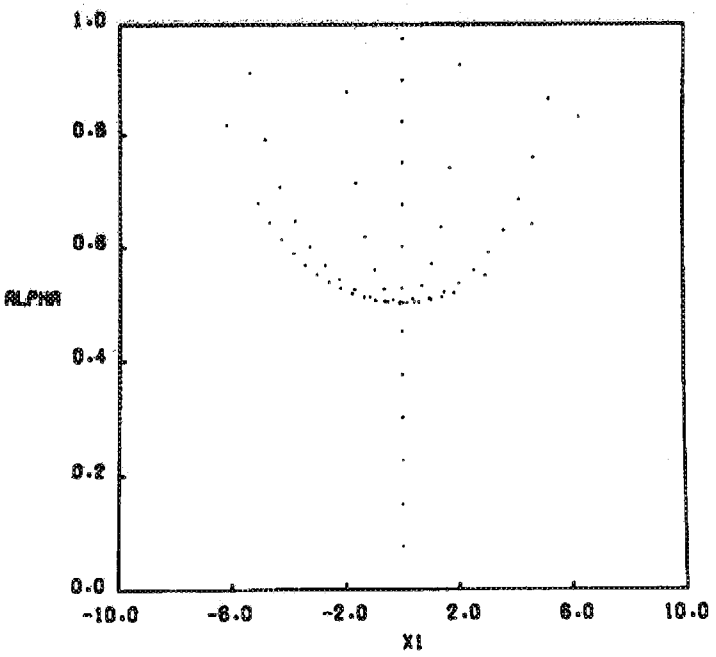

FIG. 4. The parameter is plotted versus the first component of the solution of Eq. (4.3). A graph of the third component x_3 is identical, thus not shown.

tration consists of two time stable branches connected by an unstable one. In general, we study the solution surface as a function of key membrane parameters as suggested by experiments and the developing theory of the countercurrent concentrating mechanism for urine formation.

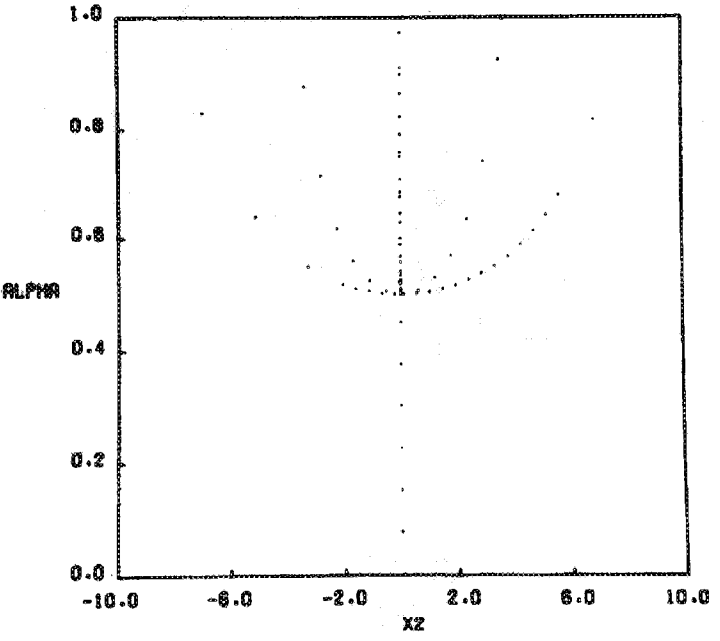

FIG. 5. The parameter is graphed against the second component x_2 of the solution.

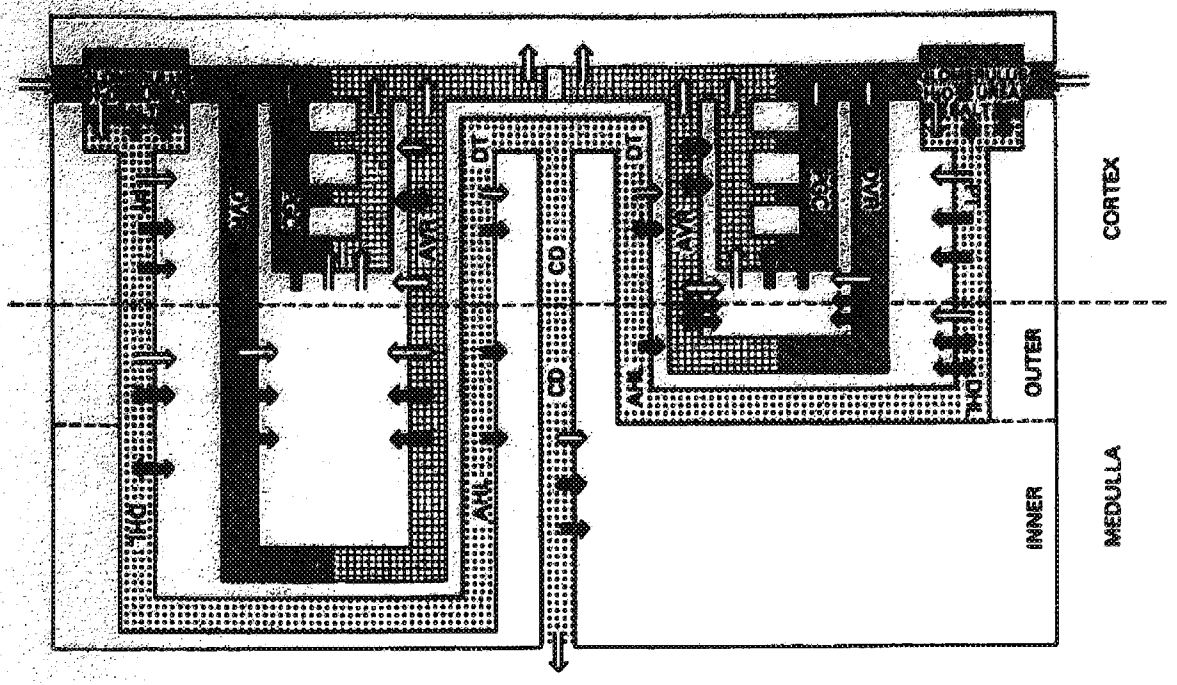

FIG. 6. A schematic diagram of a model of the mammalian kidney described in Appendix B is shown. Open arrows indicate water movement; solid arrows signify salt movement, and slashed arrows signify urea movement.

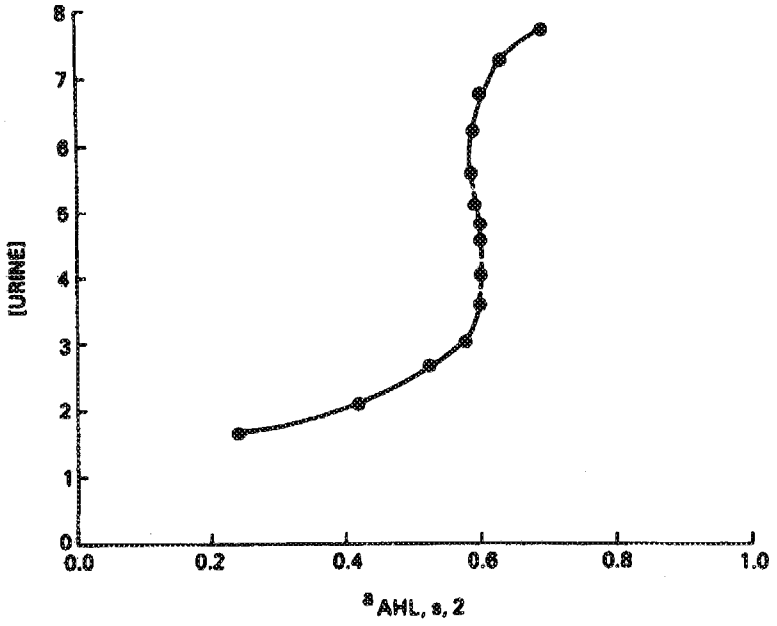

FIG. 7. The total concentration of the urine is shown as a function of the maximum rate of salt transport from the thick ascending limb of Henle in the long nephrons. The dashed part of the curve is the time unstable middle branch.

APPENDIX A

A description of each of the commands available in CONKUB, as well as a description of parameters that may be set by the user and their default values, is given in Figs. A1 and A2.

APPENDIX B

Consider a model of the mammalian kidney [25, 26] consisting of many nephrons. Each nephron is modeled as a separate nephrovascular unit [14] with water and solute exchange between nephron and vascular tubules through a common interstitial space. See the diagram shown in Fig. 6. The differential equations that describe water and solute movement in each tubular segment [13] are as follow:

$$\frac{\partial}{\partial \xi}\left(C_{ik}F_{iv} - D_{ik}\frac{\partial C_{ik}}{\partial \xi}\right) + J_{ik} + \frac{\partial}{\partial t}(A_i C_{ik}) = A_i s_{ik} \quad \text{(solute conservation)}, \tag{B.1}$$

$$\frac{\partial F_{iv}}{\partial \xi} + J_{iv} + \frac{\partial A_i}{\partial t} = 0 \quad \text{(volume conservation)}, \tag{B.2}$$

$$\frac{\partial P_i}{\partial \xi} + R_i F_{iv} = 0 \quad \text{(equation of motion)}, \tag{B.3}$$

for $0 \leqslant i \leqslant I$ tubes, $1 \leqslant k \leqslant K$, solutes, where ξ is the axial distance along the tube; $0 \leqslant \xi \leqslant l_i \leqslant 1$; l_i is the length of the ith tube; $0 \leqslant C_{ik}$ is the concentration of the kth solute in the ith tube; F_{iv} is the axial volume flow; D_{ik} is the diffusion coefficient of the kth solute in the ith tube; J_{ik} is the transmembrane solute flux; t is time; A_i is the cross-sectional area of the tube; s_{ik} is the average net rate at which the kth solute is being produced or destroyed by physical or chemical reaction; J_{iv} is the transmembrane volume flux (which is assumed to be approximately equal to the water flux); P_i is the hydrostatic pressure; and R_i is the resistance to flow.

Transmural volume and solute flux is defined as follows:

$$J_{iv} = h_{iv}\left[\sum_k RT(C_{qk} - C_{ik})\,\sigma_{ik} + P_i - P_q\right], \tag{B.4}$$

and

$$J_{ik} = h_{ik}(C_{ik} - C_{qk}) + (1 - \sigma_{ik})\,J_{iv}(C_{ik} + C_{qk})/2 + \frac{a_{ik}}{1 + b_{ik}/C_{ik}}, \tag{B.5}$$

```
**********************************************************************
   CONKUB ALGORITHM FOR PATH FOLLOWING       VERSION 042683
**********************************************************************

YOU ARE IN INITIALIZATION MODE.
VALID COMMANDS ARE:
     SB      PB      ME      GO
     SV      PV      RE      SK
     RB      PN      HE
     RV      CP      ST
     SF      DT      CM
WHEN THEY ARE NEEDED, YOU WILL BE ASKED FOR VALUES FOR:
  NVBL---NUMBER OF VARIABLES
  NPAR---NUMBER OF PARAMETERS
TYPE XX? FOR INFO ON COMMAND XX.

YOU ARE IN COMMAND CONTROL MODE.  VALID COMMANDS ARE:
   LS---LEAVE THE VARIABLE AS IT IS
   AI---ASSIGN EACH VARIABLE(I) INDIVIDUALLY
   SB---INFORMATION ON VARIABLES AND DEFAULT VALUES
   ST---RETURN TO INITIALIZATION MODE OR RUN MODE.
   ONCE IN THE AI MODE:
      LS---LEAVE THE VARIABLE THE SAME
      ST---GO ON TO THE NEXT VARIABLE.
   ENTER THE VALUE ITSELF IF ALL VARIABLES(I) ARE TO HAVE
   THE SAME VALUE.

YOU ARE IN RUN MODE.
VALID COMMANDS ARE:
     SB      PB      ME      GO
     RB      PV      RE      TU
     SF      PN      HE      LE
     SK      CP      ST      TP
             DT      CM      QV
TYPE XX? FOR INFO ON COMMAND XX.

      SET VECTOR---SV
YOU WILL BE ASKED TO INPUT INITIAL VALUES FOR THE VARIABLE
VECTOR X(I) AND THE PARAMETER VECTOR PAR(I).

      READ VECTOR---RV
READ IN THE START VECTORS FROM A FILE.  YOU WILL BE ASKED FOR THE
FILE NAME.  THE FIRST TWO VALUES MUST BE NVBL, THEN NPAR, ONE PER
LINE, FORMAT I6.  VECTOR VALUES WILL BE READ 5 PER LINE, FORMAT
D12.4,2X.  CONTROL COMMANDS ARE NOT VALID--INCLUDE ALL VALUES.

      SET BOUNDS---SB
YOU WILL BE ASKED TO INPUT VALUES(FORMAT D10.4,I6 OR L5) FOR:
   HH(I)---INTEGRAION STEP ALONG THE ARC LENGTH OF THE SOLUTION
           LOCUS. ---DEFAULT .1
   HMAX(I)---APPROXIMATE UPPER BOUNDS FOR INCREMENT OF X(I) IN ONE
             INTEGRATION STEP. --DEFAULT 1.
   EPS---ACCURACY DESIRED IN NEWTON ITERATIONS. CONVERGENCE CRITERION
         IS:  SUM OF THE ABSOLUTE VALUE OF THE CHANGE IN EACH
         COMPONENT OF X IS <= EPS. ---DEFAULT .0001
   NCORR---THE NUMBER OF NEWTON CORRECTIONS TO GET BACK NEAR THE
           CURVE.  IF THE CRITERION IS NOT MET, THE STEP SIZE WILL BE
           HALVED AND WILL TRY AGAIN.  THIS WILL BE REPEATED NCORR
           TIMES. ---DEFAULT 4
   NDIR(I)---DIRECTION OF CHANGE IN X(I)
            1---POSITIVE  -1---NEGATIVE. ---DEFAULT 1
   XUPP(I)---UPPER BOUNDS ON X(I). ---DEFAULT 40.0
   XLOW(I)---LOWER BOUNDS ON X(I). ---DEFAULT -40.0
   FACT---BIFURCATION CRITERION.  CRITERION IS THAT THE INNER PRODUCT
          OF PSI1 AND THE PARTIAL DERIVATIVE OF THE FUNCTION, F, WITH
          RESPECT TO A PARAMETER IS <= FACT; WHERE THE SPAN OF PSI1
          IS THE NULL SPACE OF THE DERIVATIVE OF F WITH RESPECT TO X.
          ALSO, SIZE OF STEP TAKEN ON EITHER SIDE OF A BIFURCATION IN
          SEARCH OF BIFURCATING BRANCH.
          ---DEFAULT .001
   BSW---SET TRUE FOR BIFURCATION TEST ON CHANGE OF DIRECTION;
        SET FALSE FOR BIFURCATION TEST ON CHANGE OF SIGN ONLY.
        ---DEFAULT  FALSE
   LPRNT---SET TRUE TO PRINT RESULT OF EACH NEWTON ITERATION.
           ---DEFAULT  FALSE
THE ITH PARAMETER BOUND WILL BE READ IN AS THE BOUND FOR X(NVBL+I)

      READ BOUNDS---RB
READ IN THE BOUNDS FROM A FILE.  YOU WILL BE ASKED FOR THE FILE
NAME.  THE FIRST TWO VALUES MUST BE NVBL AND NPAR, ONE PER LINE,
FORMAT I6.  ALL INTEGER VALUES WILL BE IN FORMAT I6; REAL VALUES WILL
BE IN FORMAT D12.4; LOGICAL VALUES IN L5.  ENTER REAL VECTOR VALUES
5 PER LINE, INTEGER VECTOR VALUES 10 PER LINE, 2X IN BETWEEN VALUES.
ENTER SCALAR VALUES ONE PER LINE.  CONTROL COMMANDS ARE NOT VALID---
INCLUDE ALL VALUES.  VALUES WILL BE READ IN THE SAME ORDER AS SB.

      HELP---HE

      STOP---ST
IF IN INITIALIZATION MODE, COMPLETES THE RUN AND RETURNS TO THE
MONITOR.  IF IN RUN MODE, RETURNS TO INITIALIZATION MODE.
```

FIGURE A1.

```
      PRINT BOUNDS---PB
PRINT OUT THE CURRENT VALUES OF THE BOUNDS.

      PRINT VECTOR---PV
PRINT OUT THE VECTOR OF VARIABLES AND PARAMETERS.

      PRINT NORM---PN
PRINT OUT THE NORM OF THE VARIABLE VECTOR AND PRINT OUT THE
PARAMETERS THAT ARE BEING FOLLOWED

      GO
IN INITIALIZATION MODE DOES INITIAL NEWTON ITERATIONS TO GET NEAR
A ROOT.
IN THE RUN MODE---GO NP--- CALCULATE THE NEXT NP POINT(S).
            INITIAL DEFAULT FOR NP IS 1.  THEN DEFAULT IS PREVIOUS NP.

     RESTART---RE
READ IN A PREVIOUSLY STORED POINT CREATED BY THE MEMORY COMMAND.
YOU WILL BE ASKED FOR THE FILE NAME.
IF YOU WISH TO GET OUT OF THIS COMMAND, INPUT ST.

     MEMORY---ME
STORES THE CURRENT POINT AND STATUS IN A FILE WITH THE NAME OF YOUR
CHOICE.  IF YOU WISH TO GET OUT OF THIS COMMAND, INPUT ST.
A MEMORY FILE WILL AUTOMATICALLY BE PRODUCED FOR POINTS ON
BIFURCATED BRANCHES.

      CHANGE PRINT---CP

YOU WILL BE GIVEN THE OPTION TO PRINT OUT THE NORM, THE VECTOR,
OR BOTH.

      SET FOLLOWING PARAMETERS---SF NPF
SPECIFY WHICH PARAMETER(S) TO FOLLOW, WHERE NPF ( <=5 )IS THE NUMBER
OF PARAMETERS. INDICES OF THE FOLLOWING PARAMETERS WILL BE
STORED IN IFOLO(I). IF IFOLO IS NOT INITIALIZED ALL PARAMETERS
MAY BE FOLLOWED. IF NPF=NPAR IFOLO WILL BE SET AUTOMATICALLY.
---DEFAULT VALUE FOR NPF = 1.
LS (LEAVE SAME) AND ST(STOP) ARE VALID, HOWEVER THEY WILL BE
IGNORED IF AN ERROR OCCURS.

      COMMENT---CM NL
INSERT A COMMENT IN THE SESSION LOG, WHERE NL IS THE NUMBER OF LINES.
EACH LINE MAY BE 75 CHARACTERS LONG.---DEFAULT FOR NL = 1.

     DATA---DT
INPUT REAL VALUES FOR RDATA(1)-RDATA(5), AND INTEGER VALUES FOR
IDATA(1)-IDATA(5), FOR OPTIONAL USE IN THE FUNCTION SUBROUTINE.
TO USE THESE ARRAYS, THE FUNCTION SUBROUTINE MUST HAVE THE STATEMENT:
COMMON/IDATA/RDATA(5),IDATA(5)

     TURN---TU NP
FOLLOW THE PATH IN THE OPPOSITE DIRECTION.
NP IS THE NUMBER OF POINTS. ---INITIAL DEFAULT = 1.  THEN DEFAULT
BECOMES THE PREVIOUS VALUE FOR NP.

     LEVEL STEPS---LE NP
FOLLOW X(K) IN LEVEL STEPS ALONG THE CURVE.
NP IS THE NUMBER OF POINTS. ---INITIAL DEFAULT = 1; SUBSEQUENTLY THE
DEFAULT IS THE PREVIOUSLY USED VALUE;
INITIAL DEFAULT ON INCREMENT = HMAX;
INITIAL DEFAULT ON ITERATIONS = 10.
IF K CHANGES OR ANY ERROR OCCURS, LEVEL MODE WILL STOP.

      TARGET POINT---TP
FIND THE SOLUTION FOR A PARTICULAR VALUE OF X(K).  YOU WILL
BE ASKED FOR THE VALUE, TRGPT.  IF K CHANGES OR ANY ERROR OCCURS,
TARGET POINT MODE WILL STOP.

      OUTPUT CURRENT VECTOR---OV
STORES THE CURRENT POINT IN A BINARY FILE WITH A NAME OF YOUR CHOICE.
EACH RECORD MAY BE READ AS FOLLOWS:
      READ(FILENAME)(X(I),I=1,NXTOT)
WHERE NXTOT=NVBL+NPAR.  THIS IS USEFUL FOR SAVING DATA FOR PLOTTING.

      SET K---SK
RESTRICTS FOLLOWING TO VARIABLE K, FOR  0 < K <= NXTOT.
YOU WILL BE ASKED FOR K.
K = 0, FREES CONKUB TO SELECT K (DEFAULT).
TURNING POINT OR BIFURCATION CALCULATIONS WILL FREE THE SELECTION OF K.
```

FIGURE A2.

for $1 \leqslant i \leqslant I$, $1 \leqslant k \leqslant K$, where h_{iv} is the hydraulic permeability coefficient of the ith tube for the kth solute; h_{ik} is the solute permeability of the ith tube for the kth solute; R is the gas constant; T is the absolute temperature; subscript q indicates an interstitial variable (for cortex $q = c$ and for medulla $q = 0$); σ_{ik} is the Staverman reflection coefficient of the wall of the ith tube for the kth solute. The last term in Eq. (B.5) defines the metabolically driven transport, which is assumed to obey Michaelis–Menten kinetics; a_{ik} is the maximum rate of transport, and b_{ik} is the Michaelis constant. Tubes i, $1 \leqslant i \leqslant I$, refer to the nephrons and vasculature of the model. In general, we assume that $s_{ik} \equiv 0$, for all i and k; that $A_i = \text{constant}$, for all i; and that $D_{ik} = 0$ for $1 \leqslant i \leqslant I$ and all k.

TABLE BI

Normalized Parameters[a]

Tube[b]	h_v	$R(\times 10^4)$	σ	σ_p	h_s	h_u		
G	1400	2	0	1	1	1	$R_A = 10.5 \times 10^{-4}$	
PGC	300	28.5	0	1	4	4	$R_E = 0.1 \times 10^{-4}$	
DVR1	100	2000	0	1	1000	1000	$R_0 = 250 \times 10^{-4}$	
DVR2	100	28.5	0	1	1000	1000	$D_{0s} = 1.0 \times 10^{-3}$	
CAVR	100	4.9	0	1	100	100	$D_{0u} = 1.0 \times 10^{-3}$	
AVR1	100	2000	0	1	1000	1000		
AVR2	100	24.5	0	1	1000	1000		
BC	0	0	1	—	0	0		
PT	—	7	—	—	—	—	$B = 0.5$	
DHL	50	10	1	—	0	0		
AHL1	0	10	1	—	0.05		$a = 1.3$,	$b = 0.1$
AHL2	0	10	1	—	0.05, 0.85[c]		$a = 0.6, 0$,[c]	$b = 0.1$
DN1	0.2	6	1	—	0	0	$a = 0.45$,	$b = 1.0$
DN2	0.2	6	1	—	0	0	$a = 0.3$,	$b = 1.0$
CD	0.44	6	1	—	0	0, 0.02[d]		
RP	0.05	0	1	—	0	0	$E = 0$.	

[a] h_v, hydraulic permeability; R, resistance to flow; σ, Staverman reflection coefficient for filtered solutes; σ_p, reflection coefficient for large solute not filtered; h_s, salt permeability; h_u, urea permeability; R_A, flow resistance afferent to glomerulus; R_E, flow resistance efferent to glomerulus; R_0, resistance to flow in the interstitium; D_{0s}, diffusion constant for salt in the interstitium; B, fraction of filtrate reabsorbed in the proximal tubule; a, maximum rate of transport; b, Michaelis constant; E, fraction of collecting duct outflow entering renal pelvis.

[b] G, glomerulus; PGC, post glomerular capillary; DVR1, descending vas rectum for first (cortical) nephrovascular unit; CAVR, cortical ascending nephrovascular unit; AVR2, ascending vas rectum for second (juxta-medullary) nephrovascular unit; BC, Bowman's capsule; PT, proximal tubule; DHL, descending loop of Henle's limb, AHL, ascending loop of Henle's limb; DN, distal nephron; CD, collecting duct; RP, renal pelvis.

[c] The first value refers to the outer medulla where $0 \leqslant x \leqslant 0.5$; the second refers to the inner medulla, where $0.6 \leqslant x \leqslant 1$. For $0.5 < x < 0.6$ the value varies linearly.

[d] The first value holds for $0 \leqslant x \leqslant 0.4$; the second holds for $0.6 \leqslant x \leqslant 1$. For $0.4 < x < 0.6$ the value varies linearly.

TABLE BII

Normalized Boundary Values

$[\text{NaCl}]_{\text{arterial}}$	1.0
$[\text{Urea}]_{\text{arterial}}$	0.05
$[\text{Large proteins}]_{\text{arterial}}$	0.0038
P_{arterial}	1.3×10^{-2}
P_{venous}	1.0×10^{-3}
P_{bladder}	1.44×10^{-3}

In the medullary region, water and mass conservation require that

$$J_{0v}(\xi) = -\sum_i J_{iv}(\xi) \tag{B.6}$$

and

$$J_{0k}(\xi) = -\sum_i J_{ik}(\xi) \tag{B.7}$$

for $0 \leqslant \xi \leqslant 1$, $1 \leqslant k \leqslant K$. In the cortex

$$J_v \equiv \sum_i \int_0^{l_i} J_{iv}(\xi)\, d\xi = -F_{0v}(0) \tag{B.8}$$

and

$$J_{ck} = -V_c \frac{\partial C_{ck}}{\partial t} - F_{0k}(0), \tag{B.9}$$

where V_c is the volume of the cortical interstitium and $0 \leqslant k \leqslant K$.

Equations (B.1)–(B.3) are discretized using a scheme that is centered in space and backward in time [13]. The interstitial variables: *Pg*, *Cgk* and *Fgv*; the boundary values for each nephron: $F_{Gv}(1)$, P_{BC} and $F_{DVRv}(1)$; plus $P_{CD}(1)$ and $F_{RPv}(n_L + 1)$ are the coordinates used for continuation. A connected component of steady state solutions is shown in Fig. 7, the values of the model parameter used are given in Table BI, and Table BII contains the boundary data for this model that has a ratio of three short nephrons for every long one.

Appendix C

Figures C1–C3 illustrate a portion of the session dialog for the solution of the first sample problem given in Section 4. For brevity, we show the calculation of the first three sample points of Figs. 1–3 beginning at $\alpha = 0$ and the computation at the

```
15:39     15-Feb-84
*****************************************************************
   CONKUB ALGORITHM FOR PATH FOLLOWING          VERSION 122983
*****************************************************************
TYPE HE FOR HELP.
CM
*****Use the SV command to enter initial values for the variables and parameters.
SV
X    =
  0.00000000D+00      0.00000000D+00      0.00000000D+00

PAR  =
  0.00000000D+00

CM
*****Use the SB command to initialize the bounds and control parameters.
SB
HH   =
  2.50000000D-01      2.50000000D-01      2.50000000D-01      1.00000000D-01

HMAX =
  2.50000000D-01      2.50000000D-01      2.50000000D-01      3.00000000D-01

EPS  =  1.00000000D-04
NDIR =
     1         1         1         1

XUPP =
  1.00000000D+01      1.00000000D+01      1.00000000D+01      1.10000000D+00

XLOW =
 -1.00000000D+00     -1.00000000D+00     -1.00000000D+00     -1.00000000D-01

FACT =  1.00000000D-03
BSW  =     F
LPRNT=     F
CM
*****Use the HE command to save the initial data.
HE
START
CM
*****Begin the computation.
GO
THERE ARE  3 VARIABLES
THERE ARE  1 PARAMETER(S)
HH   =
  2.50000000D-01      2.50000000D-01      2.50000000D-01      1.00000000D-01

HMAX =
  2.50000000D-01      2.50000000D-01      2.50000000D-01      3.00000000D-01

EPS  =  1.00000000D-04
NCORR=     5
NDIR =
     1         1         1         1

XUPP =
  1.00000000D+01      1.00000000D+01      1.00000000D+01      1.10000000D+00

XLOW =
 -1.00000000D+00     -1.00000000D+00     -1.00000000D+00     -1.00000000D-01

FACT =  1.00000000D-03
BSW= F
LPRNT= F
IFOLO=     1
RDATA=
  0.00000000D+00      0.00000000D+00      0.00000000D+00      0.00000000D+00      0.00000000D+00

IDATA=
     0         0         0         0         0

X    =
  0.00000000D+00      0.00000000D+00      0.00000000D+00

PAR  =
  0.00000000D+00
```

FIGURE C1.

```
THE VECTOR NORM IS:   0.0000D+00; PAR(  1)=  0.0000D+00
   CONKUB :  NEWTON ITER   1 ERR =  0.000D+00,   ||Xnew-Xold|| = 0.000D+00
CONKUB:  NEWTONS CONVERGED;   K=  3
CM
XXXXSave data in a form suitable for plotting using the OV command.
OV
LTHOV
CM
XXXXContinue the calculation.
GO   1
THE VECTOR NORM IS:   2.6544D-01; PAR(  1)=  5.5194D-02
   CONKUB :  NEWTON ITER   3 ERR =  6.643D-07,   ||Xnew-Xold|| = 5.610D-07
CONKUB:  NEWTONS CONVERGED;   K=  1

DET= 0.0898E2XX   4     K=  1
PV
X    =
  1.4118887D-01     1.1853171D-01     9.0685825D-02


PAR  =
  5.5194241D-02
OV
CM
XXXXChange the form of the output typed at the terminal using the CP command.
CP
CM
XXXXContinue the calculation.
GO   1
X    =
  4.3382268D-01     2.2820534D-01     1.0984143D-01


PAR  =
  2.1059100D-01

   CONKUB :  NEWTON ITER   4 ERR =  3.362D-08,   ||Xnew-Xold|| = 4.623D-08
CONKUB:  NEWTONS CONVERGED;   K=  1

DET= 0.1241E2XX   4     K=  1
         .
         .
         .
         .       (For brevity we omit most of the computation and
         .         show the last two points computed on the curve.)
         .
         .
         .
GO   1
X    =
  3.7470761D-01     2.5266946D+00     4.3248498D-01


PAR  =
  9.9995987D-01

   CONKUB :  NEWTON ITER   1 ERR =  7.829D-16,   ||Xnew-Xold|| = 3.081D-16
CONKUB:  NEWTONS CONVERGED;   K=  3

DET= 0.1323E2XX   4     K=  3
OV
CM
XXXXLook at the current bounds using the PB command.
PB
THERE ARE  3 VARIABLES
THERE ARE  1 PARAMETER(S)
HH   =
  1.0000000D-03     1.0000000D-03     1.0000000D-03     1.0000000D-03
=
HMAX =
  1.0000000D-04     1.0000000D-04     1.0000000D-04     1.0000000D-04

EPS  =  1.0000000D-08
NCORR=     5
NDIR =
=    1        -1        1         1


XUPP =
  1.0000000D+01     1.0000000D+01     1.0000000D+01     1.1000000D+00

XLOW =
 -1.0000000D+00    -1.0000000D+00    -1.0000000D+00    -1.0000000D-01
```

FIGURE C2.

```
FACT =   1.00000000D-03
BSW= F
LPRNT= F
IFOLO=      1
RDATA=
  0.00000000D+00      0.00000000D+00      0.00000000D+00      0.00000000D+00      2.54000000D+02


IDATA=
      0          0          0          0          0


CM
****Use the SB command to reduce the step size allowed.
SB
HH    =
  5.00000000D-04   5.00000000D-04      5.00000000D-04      5.00000000D-04


HMAX =
  5.00000000D-05      5.00000000D-05      5.00000000D-05      5.00000000D-05


GO    1
X     =
  3.74726268D-01      2.52667688D+00      4.32544178D-01


PAR   =
  9.99997528D-01

    CONKUB :  NEWTON ITER    2 ERR =  6.387D-16,    ||Xnew-Xold|| = 4.763D-16
CONKUB:  NEWTONS CONVERGED;    K=  3

DET= 0.1323**2**   4      K=  3
OV
ST
ST
```

FIGURE C3.

last two points approaching $\alpha = 1$. See Appendix A for a detailed description of the commands and other mnemonics used.

ACKNOWLEDGMENTS

The conversational program is driven by a program written by Carol Whitney. Its functions implement many useful ideas from SCOUT, a simplicial continuation program due to Jürgens *et al.* [8]. The continuation algorithm is based on that by Kubiček [11] and the implementation of Bunow and Kernevez [2].

REFERENCES

1. E. ALLGOWER AND K. GEORG, *SIAM Rev.* **22** (1980), 28.
2. B. BUNOW AND J. KERNEVEZ, *in* "Instabilities, Bifurcation, and Fluctuations in Chemical Systems" (L. E. Reichl and W. C. Schieve, Eds.), Univ. of Texas Press, Austin, 1982.
3. S.-N. CHOW, J. MALLET-PARET, AND J. A. YORKE, *Math. Comput.* **32** (1978), 887.
4. M. G. CRANDALL AND P. H. RABINOWITZ, *J. Funct. Anal.* **8** (1971), 321.
5. M. G. CRANDALL AND P. H. RABINOWITZ, *Arch. Ration. Mech. Anal.* **52** (1973), 161.
6. E. DOEDEL, *Congr. Numer.* **30** (1981), 265.
7. K. GEORG, *SIAM J. Sci. Stat. Comput.* **2** (1981), 35.
8. H. JÜRGENS, H-O. PEITGEN, AND D. SAUPE, *in* "Analysis and Computation of Fixed Points," p. 139, Academic Press, New York, 1980.

9. R. B. KEARFOTT, *SIAM J. Sci. Stat. Comput.* **4** (1983), 52.
10. H. B. KELLER, *in* "Applications of Bifurcation Theory," p. 359, Academic Press, New York, 1977.
11. M. KUBIČEK, Algorithm 502, *ACM Trans. Math. Software* **2** (1976), 98.
12. R. MEJIA, R. B. KELLOGG, AND J. L. STEPHENSON, *J. Comput. Phys.* **23** (1977), 53.
13. R. MEJIA AND J. L. STEPHENSON, *J. Comput. Phys.* **32** (1979), 235.
14. R. MEJIA AND J. L. STEPHENSON, Numerical Methods for Engineering, *G.A.M.N.I.* **2** (1980), 1003.
15. R. MEJIA AND J. L. STEPHENSON, *Math. Biosci.* **68** (1984), 279.
16. G. MOORE AND A. SPENCE, *SIAM J. Numer. Anal.* **17** (1980), 567.
17. J. J. MORÉ AND M. Y. COSNARD, *ACM Trans. Math. Software* **5** (1979), 64.
18. J. J. MORÉ AND M. Y. COSNARD, *ACM Trans. Math. Software* **6** (1980), 240.
19. A. P. MORGAN, *ACM Trans. Math. Software* **9** (1983), 1.
20. H-O. PEITGEN AND M. PRÜFER, *in* "Springer Lecture Notes in Mathematics, Vol. 730," p. 326, 1980.
21. H-O. PEITGEN AND K. SCHMITT, *in* "Springer Lecture Notes in Mathematics, Vol. 878," p. 275, 1981.
22. W. C. RHEINBOLDT, *SIAM J. Numer. Anal.* **17** (1980), 221.
23. W. C. RHEINBOLDT AND J. V. BURKHARDT, *ACM Trans. Math. Software* **9** (1983), 215.
24. W. C. RHEINBOLDT AND J. V. BURKHARDT, *ACM Trans. Math. Software* **9** (1983), 236.
25. J. L. STEPHENSON, *Biophys. J.* **16** (1976), 1273
26. J. L. STEPHENSON, R. MEJIA, AND R. P. TEWARSON, *Proc. Natl. Acad. Sci. U.S.A.* **73** (1976), 252.
27. R. P. TEWARSON, J. L. STEPHENSON, AND L. L. JUANG, *J. Math. Anal Appl.* **63** (1978), 439.
28. L. T. WATSON, *SIAM J. Numer. Anal.* **16** (1979), 394.
29. L. T. WATSON AND D. FENNER, *ACM Trans. Math. Software* **6** (1980), 252.